Real Anger Detection using Neural Network with Compression

Weifan Jiang[1]

[1] Research School of Computer Science, Australian National University, Australia
[2] u6683698@anu.edu.au

Abstract. Emotional judgment has long been an object of interest by scientists. Human emotions are mostly expressed by the details of people's facial expressions. Anger, one of the most common human emotions, has been found to be associated with a variety of physiological activities(Lu et al., 2017) such as pupillary response patterns, and achieve 95% accurate base on the physiological signals [2]. Here I introduce a RNN model with compression, Based on anger dataset which is used by [2] Lu et al., the model achieved 97% accurate. After Distinctiveness network reduction, it also achieved 93% accuracy.

Keywords: Neural Networks · Classification · Network Compression.

1 Introduction

Research on emotion has been a subject for hundreds of years. Emotion detection has been put in practice for years, like Human-computer interaction, security, robotics, medicine, communications, and automotive. For example, analysis of customer sentiment in a store. That is, the customers in the mall or store are captured through the camera, their facial expressions are analyzed, and then the emotional information of the customers is further interpreted, to analyze the customers' satisfaction in the mall. Most emotion recognition is based on changes in facial features like eyebrows, eyes, lips. Facial expression is the most direct and effective mode of emotion recognition. The limitation of the model is the facial expressions may not express people's emotions, which means being asked to smile may not actually be happy. Hence the physiology-based recognition model.

1.1 Physiological Signals

Physiological signals such as blood pressure, blinking, and brain waves can reflect the state of a person to some extent, such as the level of fatigue, happiness. this kind of signals is not influenced by the outside world, but the true expression of people's behavior and feelings. Zakir et al. use this natural advantage to collect physiological signals such as galvanic skin response (GSR), blood volume pulse (BVP), and pupillary response (PR) to detect whether your smile is posed (ask to act) or real(elicited), and test on k-nearest neighbor, support vector machines, Neural network, and other models [9]. Ping et al. utilise four kinds of physiological signals, SC (skin conductivity), EMG(electromyogram), ECG(electrocardiogram), RSP(respiratory changes) to recognize emotions [5]. Deep learning is adopted as well. TSception(Yi et al.,2020) is a deep learning framework for emotion detection, using spatial and temporal convolutional layers to process EEG(electroencephalogram), which ensure that discriminative representations and channel domains are learned [3]. From above, we know that advantages of using physiological signals.Therefore, the dataset of this experiment extract and processed from Physiological Signals. From the data acquisition method, we can see that it contains a lot of time spacial information. Therefore, the corresponding use of the network to extract features on the timing is necessary.

1.2 Long Short Term Memory

The recurrent neural network is that takes sequence data as input, recurses in the sequence evolution direction and links all nodes (recurrent units) in a chain, and the same to LSTM(Long Short Term Memory) network. LSTM networks, an improved recurrent neural network, can not only process the sequence changing data but also solve the problem of long-distance dependency. The hidden layer of the original RNN has only one state, which is very sensitive to short-term input. In LSTM module another state is added, to hold the long-term state. This state called Cell state and controlled by 3 gates — output gate, input gate, and output gate. Forget gate is to control what needs to be left and what needs to be forgotten from the previous node to current node. The function of input gate is selectively "remember" the input of the node at current sequence. Output gate controls the output of the node. Control the transmission state through the gated state, remember the need to remember for a long time, and forget the unimportant information. So the temporal information in pupillary response is extracted. [8]

1.3 Neural Network Reduction

Nowadays, deep learning has become one of the most mainstream branches of machine learning.one of the disadvantage of the deep neural network require a lot of computation and take up a lot of memory. The deeper the layers and the more parameters the neural network has, the more precise the results, which means more computing resources are consumed. Most of them are trained and get great performance on a high-performance computer or GPU cluster. If you simply deploy to an embedded system, or to a mobile system, power consumption, time delays, etc., can be a problem. Simplify the model to reduce computation and storage footprint became the focused area of many scientists.

There are many parameters in a neural network, but some of them contribute little to the final output result and appear redundant. As the name implies, pruning is to cut out these redundant parameters. the basic process of the technique is devising an algorithm to decide which neuron has no particular contribution to finding features. Then we could remove the relatively useless neuron in order to reduce the size of hidden layers and its' parameters. Thus, the calculation efficiency can be improved by sacrificing a small percentage of accuracy or none at all.

Neural network Pruning is not a new problem and was studied in the early 1990s. Magnitude-based pruning method (Stephen et al.,1989) proposed adding a weight decay related to its absolute value to each hidden unit in the network in order to minimize the number of hidden units [7].

Dropout [12] and DropConnect [13] using zero activation and random zero connection to reduce overfitting rather than improve computational efficiency Self-Adaptive Network Pruning(SANP) (Jinting et al.,2019) is a new technique that has been put forward recently. In each convolutional layer, the method introduced a Saliency-and-Pruning Module(SPM) and this module allows the network to predict saliency scores which is the basis of network pruning [1]. This report shows how to use Distinctiveness [4] to pruning network. Here are several attribution to measure whether a neuron should be reduction or not. Contribution analysis can scoring each hidden neurons responsibilities. For one input sample, contribution is the product of weight of hidden neuron to output neuron and activation function [11]. Some disadvantages have emerged, and contribution analysis does not adopted on pruning in practice. Sensitivity of the global error function(Karnin,1990) introduce sensitivity analysis to each synapse in a neural network and keep tracking incremental changes in synaptic weights during back propagation learning by "shadow arrays". According Sensitivity indicator to sort synapse in decrease order and pruning the neural network by discarding the last one [10] which is a neuron with the lowest Sensitivity. Hagiwara introduced an algorithm to detect worst hidden neuron by calculation "Badness factor", in 1990. This algorithm introduce a back propagated error component for each hidden neurons, and "Badness factor" is sum of these component over all patterns. Then the hidden neuron with a biggest "Badness factor" will be removed. In addition, during back propagation, the convergence is accelerated by resetting every the weight connected to the worst neuron to small random values [6].

Distinctiveness using vector angle to analyze the different feature extracting from different samples or patterns by hidden neurons in the same layer. Vector Angle calculation is to analyze the similarity of results of different pairs of hidden neurons upon the same input pattern. The angle between a pair of vectors is small, which means these two hidden neurons are similar functionally. Vice versa, If the functionality of a pare of vector cancels out, the large vector angle will appear and the two corresponding vectors can be removed.

2 Method

In this section, I will describe what does input data looks like, model structure, and more details about network reduction. The aim of this task is to train a classifier to detect that observers are angry or not, given the anger dataset.

2.1 Dataset

Lu et al. analyzed a physiological signal — pupillary response and applied quantitative analysis [9]. Then, two data file about pupillary response were generated, corresponding to the left and right eyes of the 20 participants respectively. In the group of 10 real anger and 10 posed anger experiment, data of each participant were recorded respectively. In my experiment, All data have been normalised according to Equation 1,

$$x_{norm} = \frac{x - \text{mean}(x)}{\max(x) - \min(x)} \tag{1}$$

where x is every element in anger dataset. In addition, some NAN data was deleted. We ended up with 390 pairs of left eye and right eye data. And each pair of data contains a rich sequential infomation, which is the reason I choose RNN with LSTM module to solve the classification problem.

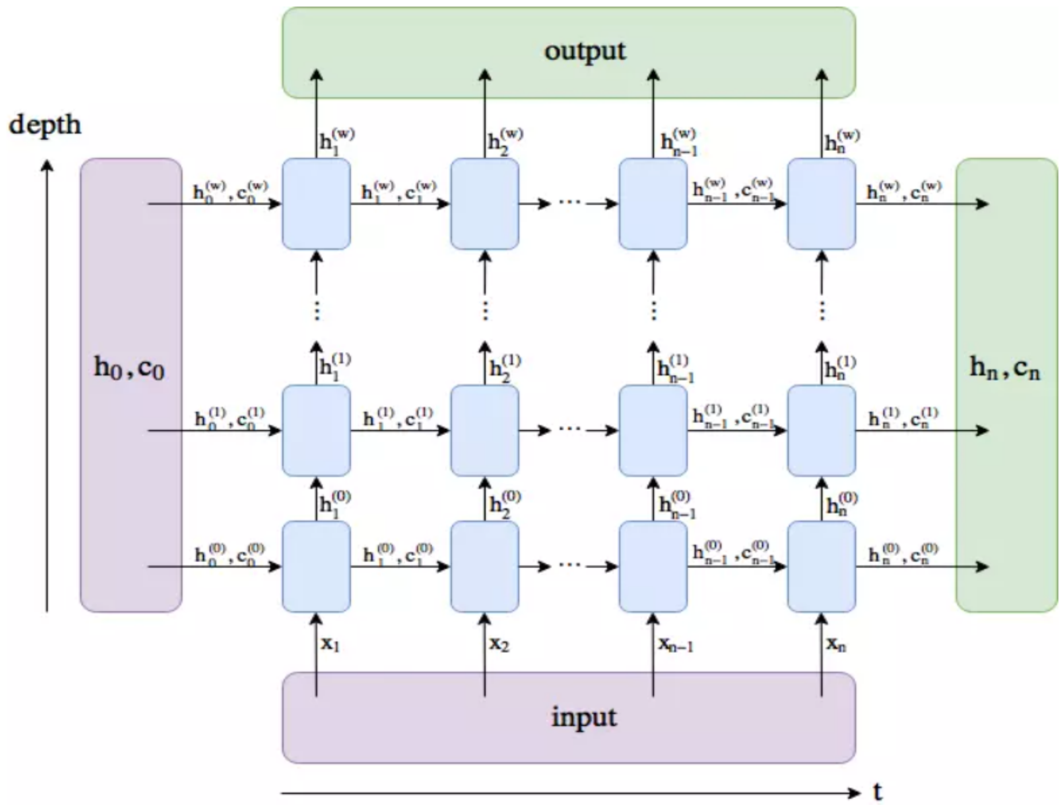


Fig. 1. Fig.1 shows the structure of first part of the model. In the figure, each blue block represents LSTM module Output block stand for $\boldsymbol{v}_i$

2.2 Network Structure

For ease of expression, I denote i-th input data as $\boldsymbol{x}_i = \langle \boldsymbol{x}_{ij} | i = 1, \ldots, M; j = 1, \ldots, N \rangle$, M equals 178 in the anger dataset, which means for every sample data $\boldsymbol{x}$ of variable length, padding it to the longest sequence M of all the data. When a single human eye is used as the input, N equals 780 and $\boldsymbol{x}_i \in \mathbb{R}^{178*2}$. If the left and right eyes of the data are formed into a pair as the input, then N equals 390 and $\boldsymbol{x}_i \in \mathbb{R}^{178*1}$. $\boldsymbol{Label}$ $\langle Label_i | i = 1, \ldots, N \rangle$ represents label of corresponding the input data.

The network is divided into two parts. The first part is to use a LSTM with 5 hidden layer and 64 hidden size to obtain the temporal information of the data, denoted as $\boldsymbol{Rnn(*)}$.

$$\boldsymbol{v}_i \in \mathbb{R}^{64} = \boldsymbol{Rnn}\left(\boldsymbol{x}_i\right) \tag{2}$$

The second part uses two fully connected layers. The first fully connected layer re-represents $\boldsymbol{o}_I$with 256 dimensional vectors, and The second fully connected layer maps this 256 representations into a 2-dimensional vector, which respectively corresponds to the categorical characteristics of The input data. Form above, the second part also kown as classification layer. Denote it as $\boldsymbol{\gamma(*)}$,

$$\boldsymbol{o}_i \in \mathbb{R}^2 = \boldsymbol{\gamma}\left(\boldsymbol{v}_i\right) \tag{3}$$

After that i use softmax function and Cross-Entropy loss to calculate error, define as

$$p_i = \operatorname{softmax}\left(\mathrm{o}_i\right) = \frac{e^{\mathrm{o}_i}}{\sum_{j=1}^{n} e^{\mathrm{o}_i}} \tag{4}$$

$$L_{\text{crossEntropy}}\left(p_i\right) = -\frac{1}{M} \sum_i \sum_l \log p_{i,l} Label_{i,l} \tag{5}$$

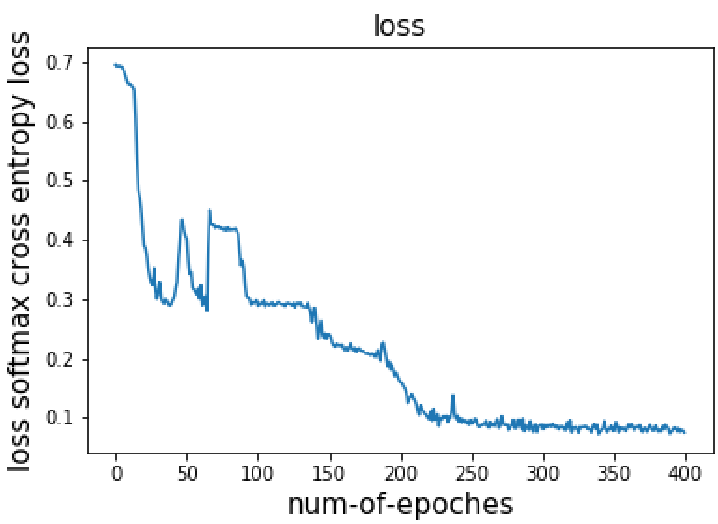
, where $l \in (1, 2)$. each element in p_i is probability that i-th pattern belongs to l class. then the task is minimize the Cross-Entropy loss.

2.3 Network reduction

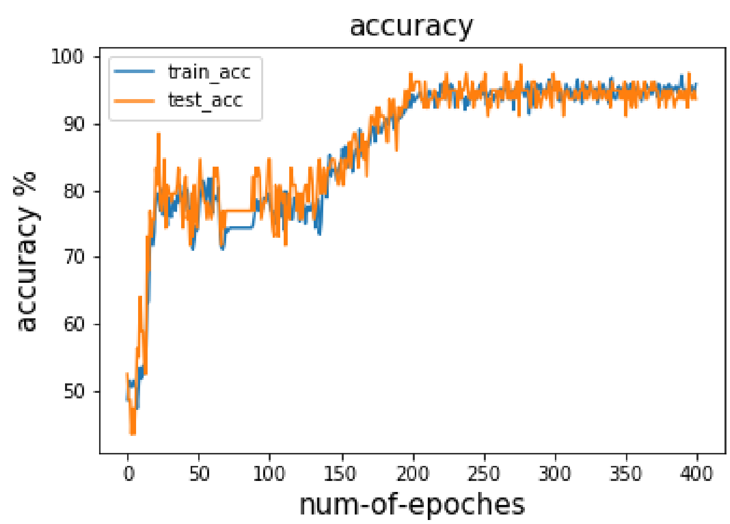
Network reduction according to distinctiveness was applied in the second part of the model. Distinctiveness [4] is based on the theory that the same hidden neuron in the same layer picks up the same features in different patterns and the different hidden neurons in the same layer extract the same features in different patterns. The functionality of each hidden neurons lies in the output vector. From the introduction part, we know that we calculate and compare the angle between each column that shows in Table 1. If the angle of a pair of vectors is less than 15°, one of the vectors should be removed and the weight vector of the removed units is added to the weight vector of the remaining one. In addition, if the angle of a pair of vectors is less than 165°, both two vectors should be removed.

	Neuron 1	Neuron 2	......	Neuron 256
Pattern 1	0.4953484	0.456046	......	0.46065
Pattern 2	0.2795404	0.202135	......	0.16054
Pattern 3	-0.246548	0.302135	......	0.00602
Pattern 4	0.1654610	-0.002135	......	-0.05405
Pattern 5	0.1345604	-0.102135	......	0.00214

Table 1. Table 1 shows sample output from sigmoid activation function and normalized with 0.5



(a) Fig 2.1



(b) Fig 2.2

Fig. 2. Fig 2.1 illustrate loss during training procedure. Fig2.2 illustrates train and test accuracy in each epochs.

3 Results and Discussion

This model was designed with four different experiments, as shown in table 2. I finally made the model with human eyes as input the main discussion object of this paper. This model was designed with four different experiments, as shown in table 2. I finally made the model with two human eyes as input the main discussion object of this paper. For each experimental group, 80% of all data were used as training data and the rest as test data. Because of the small amount of data, I did not design the validation set. In addition, Each model experienced 400 epochs, and the learning rate using dynamic change, that is 0.001 at the initial learning rate. For each 100 epoch, the learning rate was divided by 2. In the training process, Adam method is adopted to optimise the model. The model's running speed obtained on RTX2070 graphics card under Windows.

Model	Accuracy	precision	recall
PDLeft	85.1353 %	83.4650 %	84.6541%
PDRight	88.4801 %	85.6540 %	86.7894 %
PDLeft+PdRight	93.5897 %	91.4512%	92.4512%
PDLeft&PdRight	**97.0021** %	**93.1052**%	**94.5945** %

Table 2. Table 2 shows results for each model. The results for each model were averaged 10 sessions of the best trained model

Model after pruned	# of neuron pruned	Accuracy	precision	recall
PDLeft	31	78.4562%	79.6541%	80.0012%
PDRight	48	80.5243%	83.2145%	84.9241 %
PDLeft+PdRight	56	90.0152 %	89.8464%	88.3442%
PDLeft&PdRight	41	**94.2468%**	**93.4250%**	**93.7543%**

Table 3. Table 3 shows results for each pruned model. The results for each pruned model were averaged 10 sessions of the best trained model

Model	time cost	after pruned
PDLeft	0.1086s	0.1042s
PDRight	0.1021s	0.1019s
PDLeft+PdRight	0.1102s	0.1012s
PDLeft&PdRight	**0.1276s**	**0.1062s**

Table 4. Table 4 shows time cost of each model. Test the model on whole data 100 times and average the time cost.

4 Conclusion and Future Work

From the result above we can see this model works well and the accuracy of PDLeft&PdRight model reaching 97%, slightly better than Lu et al. 95%. And after network pruning the model shows slightly speed up.

For improving the anger detection model, there is some potential future work from a different aspects.

1. In terms of pruning method, Self-Adaptive Network Pruning [1] is a worthwhile approach. By adding a special designed module in each layer of the neural network, the model can be pruned during training.
2. In terms of training process, The loss function design significantly influences convergence. Different type of Loss Function would be worthwhile like Triplet Loss used in metric learning.

References

1. Chen, J., Zhu, Z., Li, C., Zhao, Y.: Self-adaptive network pruning. In: International Conference on Neural Information Processing. pp. 175–186. Springer (2019)
2. Chen, L., Gedeon, T., Hossain, M.Z., Caldwell, S.: Are you really angry? detecting emotion veracity as a proposed tool for interaction. In: Proceedings of the 29th Australian Conference on Computer-Human Interaction. pp. 412–416 (2017)
3. Ding, Y., Robinson, N., Zeng, Q., Chen, D., Chai, A.A.P., Lee, T.S., Guan, C.: Tsception: A deep learning framework for emotion detection using eeg. arXiv preprint arXiv:2004.02965 (2020)
4. Gedeon, T., Harris, D.: Network reduction techniques. In: Proceedings International Conference on Neural Networks Methodologies and Applications. vol. 1, pp. 119–126 (1991)
5. Gong, P., Ma, H.T., Wang, Y.: Emotion recognition based on the multiple physiological signals. In: 2016 IEEE International Conference on Real-time Computing and Robotics (RCAR). pp. 140–143 (2016)
6. Hagiwara, M.: Novel backpropagation algorithm for reduction of hidden units and acceleration of convergence using artificial selection. In: 1990 IJCNN international joint conference on neural networks. pp. 625–630. IEEE (1990)
7. Hanson, S.J., Pratt, L.Y.: Comparing biases for minimal network construction with back-propagation. In: Advances in neural information processing systems. pp. 177–185 (1989)
8. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)
9. Hossain, M.Z., Gedeon, T.: Classifying posed and real smiles from observers' peripheral physiology. In: Proceedings of the 11th EAI International Conference on Pervasive Computing Technologies for Healthcare. pp. 460–463 (2017)
10. Karnin, E.D.: A simple procedure for pruning back-propagation trained neural networks. IEEE transactions on neural networks **1**(2), 239–242 (1990)
11. Sanger, D.: Contribution analysis: A technique for assigning responsibilities to hidden units in connectionist networks. Connection Science **1**(2), 115–138 (1989)
12. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research **15**(1), 1929–1958 (2014)
13. Wan, L., Zeiler, M., Zhang, S., Le Cun, Y., Fergus, R.: Regularization of neural networks using dropconnect. In: International conference on machine learning. pp. 1058–1066 (2013)